Calculator with DevOps

Akanksha Shukla

MT2022008

Links

Github Repo :

https://github.com/imshukla12/calculatorDevOps

Docker Repo :

https://hub.docker.com/repository/docker/imshukla/calculator/general

Problem Statement

Create a scientific calculator program with user menu driven operations

● Square root function - √x

● Factorial function - x!

● Natural logarithm (base e) - ln(x)

● Power function – x^b

DevOps

- DevOps is the combination of cultural philosophies, practices, and tools that increases an organization's ability to deliver applications and services at high velocity: evolving and improving products at a faster pace than organizations using traditional software development and infrastructure management processes. This speed enables organizations to better serve their customers and compete more effectively in the market.
- DevOps combines development (Dev) and operations (Ops) to unite people, process, and technology in application planning, development, delivery, and operations. DevOps enables coordination and collaboration between formerly siloed roles like development, IT operations, quality engineering, and security.
- Teams adopt DevOps culture, practices, and tools to increase confidence in the applications they build, respond better to customer needs, and achieve

business goals faster. DevOps helps teams continually provide value to customers by producing better, more reliable products.

Benefits of DevOps:

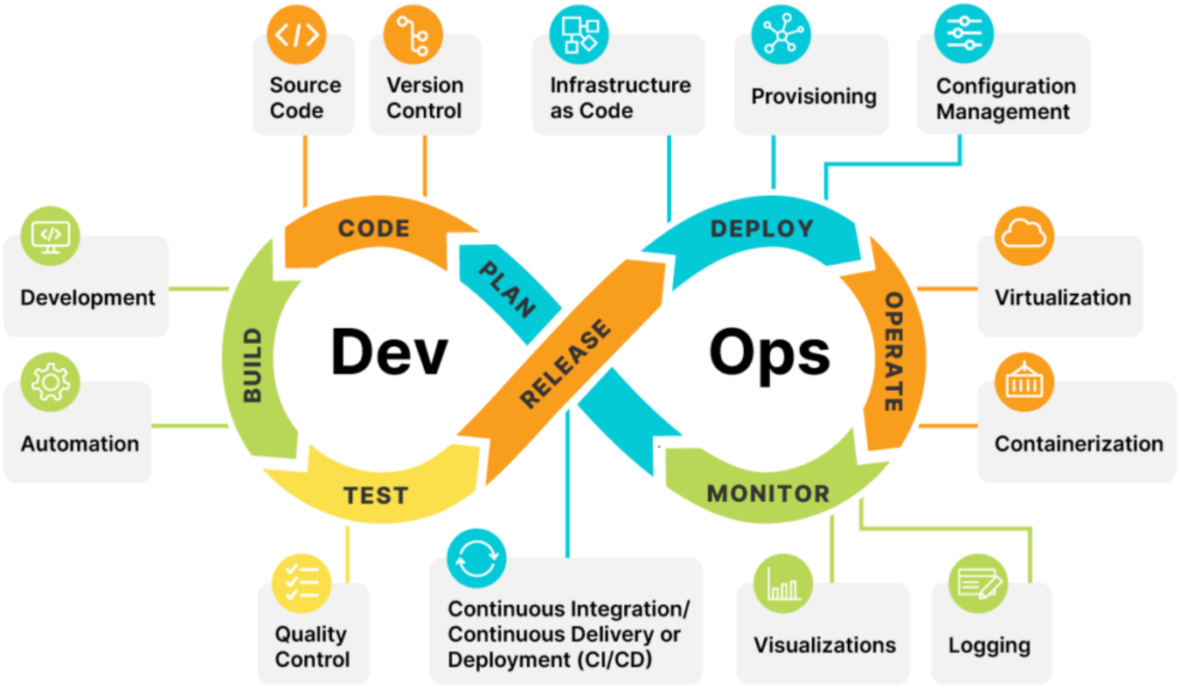
- Speed
  Move at high velocity so you can innovate for customers faster, adapt to changing markets better, and grow more efficient at driving business results. The DevOps model enables your developers and operations teams to achieve these results.

- Rapid Delivery
  Increase the frequency and pace of releases so you can innovate and improve your product faster. The quicker you can release new features and fix bugs, the faster you can respond to your customers' needs and build competitive advantage. Continuous integration and continuous delivery are practices that automate the software release process, from build to deploy.

- Reliability
  Ensure the quality of application updates and infrastructure changes so you can reliably deliver at a more rapid pace while maintaining a positive experience for end users. Use practices like continuous integration and continuous delivery to test that each change is functional and safe. Monitoring and logging practices help you stay informed of performance in real-time.

- Scale
  Operate and manage your infrastructure and development processes at scale. Automation and consistency help you manage complex or changing systems efficiently and with reduced risk.

- Security
  Move quickly while retaining control and preserving compliance. We can adopt a DevOps model without sacrificing security by using automated compliance policies, fine-grained controls, and configuration management techniques.



Why DevOps ?

Software and the Internet have transformed the world and its industries, from shopping to entertainment to banking. Software no longer merely supports a business; rather it becomes an integral component of every part of a business. Companies interact with their customers through software delivered as online services or applications and on all sorts of devices. They also use software to increase operational efficiencies by transforming every part of the value chain, such as logistics, communications, and operations. In a similar way that physical goods companies transformed how they design, build, and deliver products using industrial automation throughout the 20th century, companies in today's world must transform how they build and deliver software.

Tools used

- **Maven**: It's a Java-based application development tool that lets us add dependencies and build a jar file (a snapshot of our project) that can be run on any machine.

- **GitHub**: Helps in automation through Jenkin Integration Calculator with DevOps

- **WebHooks**: To automate the build process whenever the developer commits the code to GitHub.

● **Ngrok**: To convert the private IP address of the local machine to a public IP address to perform a webhook.

● **Jenkins**: It is used for DevOps(for Continuous Integration and Continuous Deployment portion)

● **Docker**: It is used to make images through containerization.

● **Ansible**: It automates and simplifies repetitive, complex, and tedious operations. It saves a lot of time when we install packages or configure large numbers of servers.

Steps

1. Install Java 11 and IntelliJ

2. Write your code in Maven

3. Push your code into GitHub

4. Create a repository in Docker Hub for your project

5. Write Pipeline Script in Jenkins

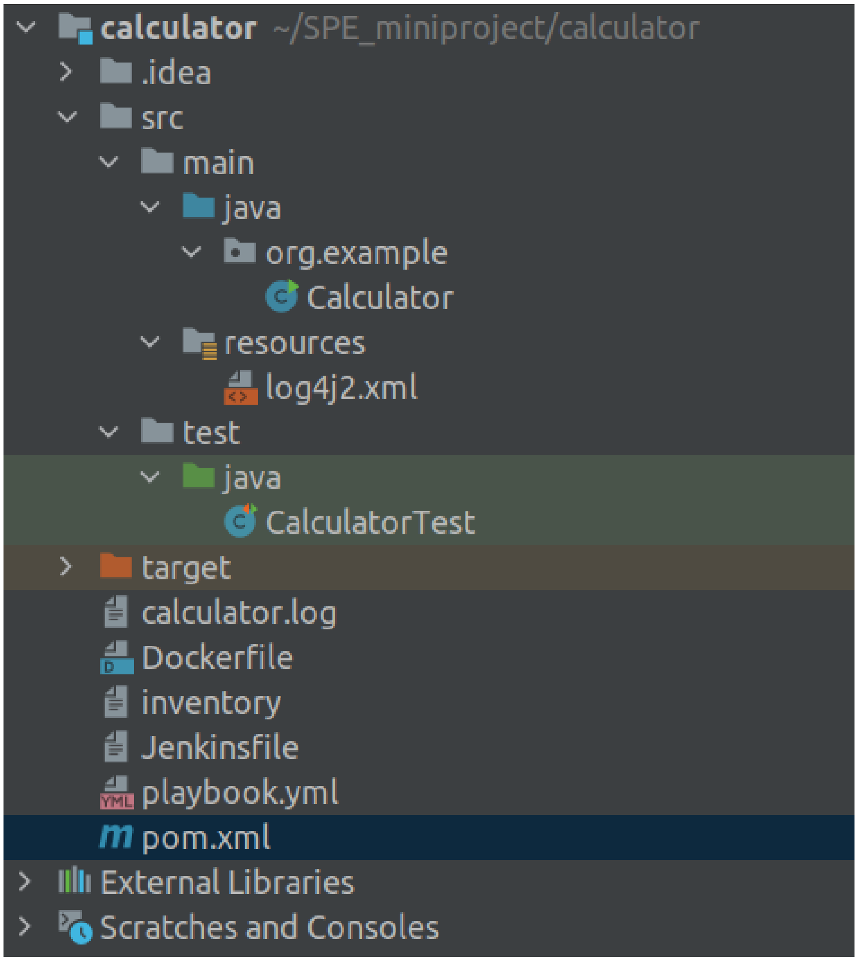
    a. Git Pull
    b. Maven build
    c. Docker Image creation
    d. Pushing Image to Docker Hub
    e. Ansible Deploy
    f. Build the project.
    g. Pull the image into the remote server.
    h. Run the image

Steps involved :

Developing the calculator program, building, and testing :

The code is developed in Java 11 and IntelliJ IDE is utilized as the development environment. JUnit is used for unit testing and log4j is used for generating logs.

The project structure will have two important folders, the main folder where we write the main code of our calculator program, and the test folder where you write the test cases for our calculator program. Along with these two folders, we have the 'pom.xml' file (It is an XML file that contains information about the project and configuration details used by Maven to build the project).

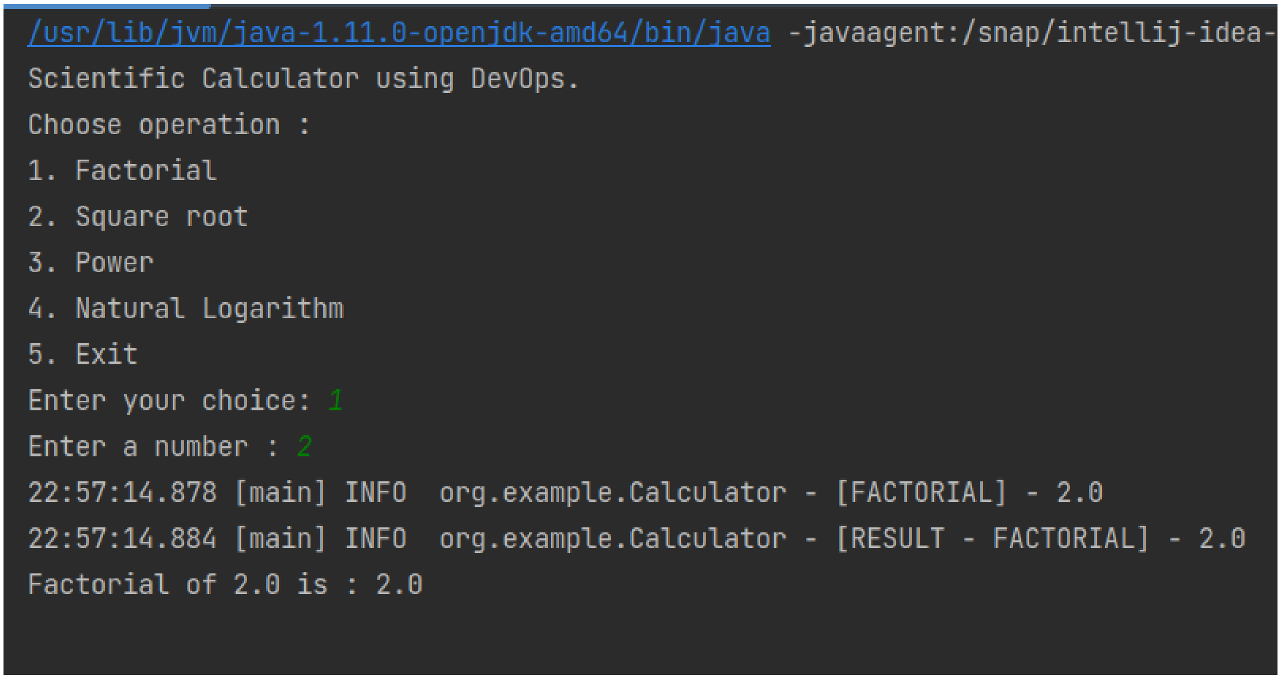


```
calculator ~/SPE_miniproject/calculator
  .idea
  src
    main
      java
        org.example
          Calculator
      resources
        log4j2.xml
    test
      java
        CalculatorTest
  target
  calculator.log
  Dockerfile
  inventory
  Jenkinsfile
  playbook.yml
  pom.xml
External Libraries
Scratches and Consoles
```

Calculator.java: The calculator program written in java contains four functions, namely:

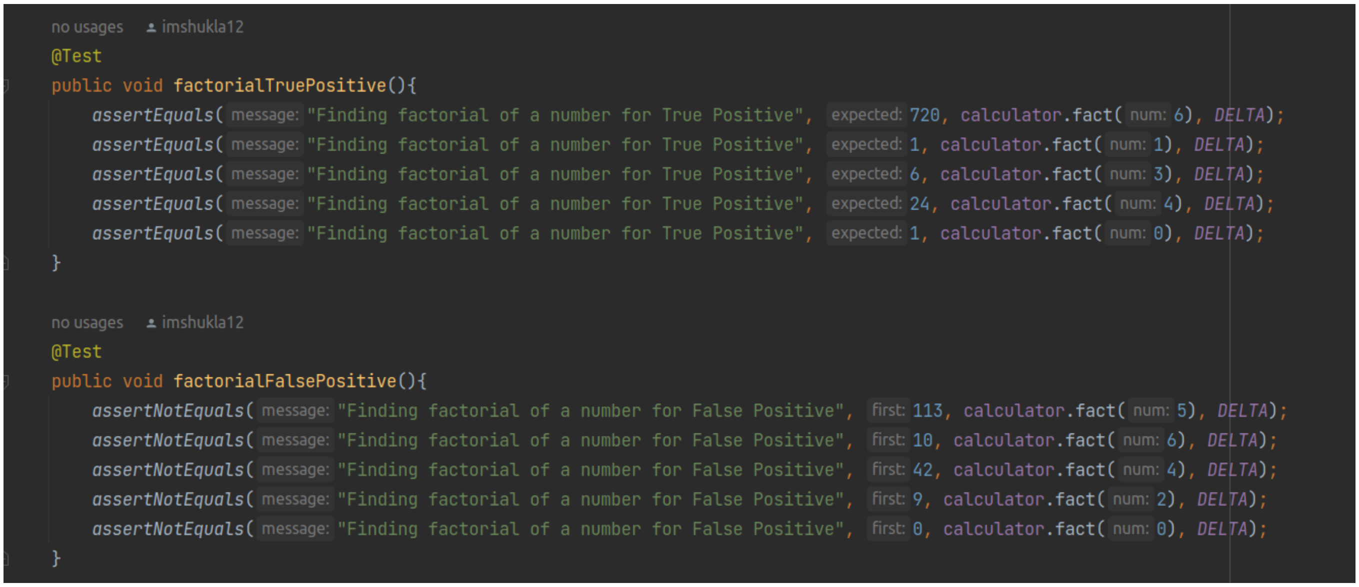
- Factorial
- Square_root
- Natural_log
- X^b

and produce the following output when run :



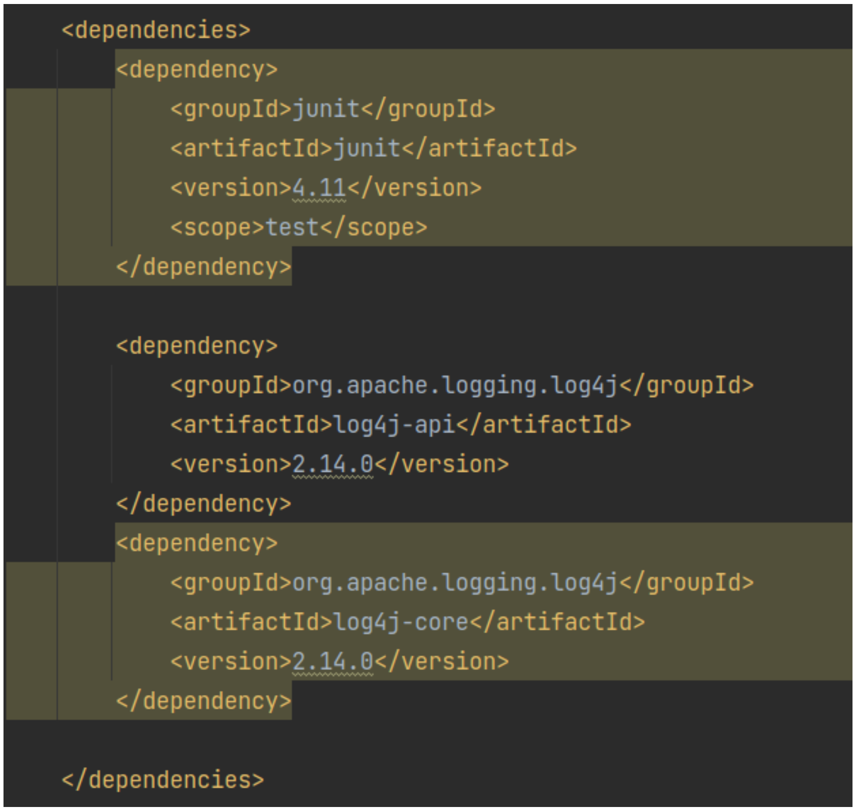
```
/usr/lib/jvm/java-1.11.0-openjdk-amd64/bin/java -javaagent:/snap/intellij-idea-
Scientific Calculator using DevOps.
Choose operation :
1. Factorial
2. Square root
3. Power
4. Natural Logarithm
5. Exit
Enter your choice: 1
Enter a number : 2
22:57:14.878 [main] INFO  org.example.Calculator - [FACTORIAL] - 2.0
22:57:14.884 [main] INFO  org.example.Calculator - [RESULT - FACTORIAL] - 2.0
Factorial of 2.0 is : 2.0
```

CalculatorTest.java: Contains true and false positive test cases used to test the code when we build the project. It is performed using JUnit.

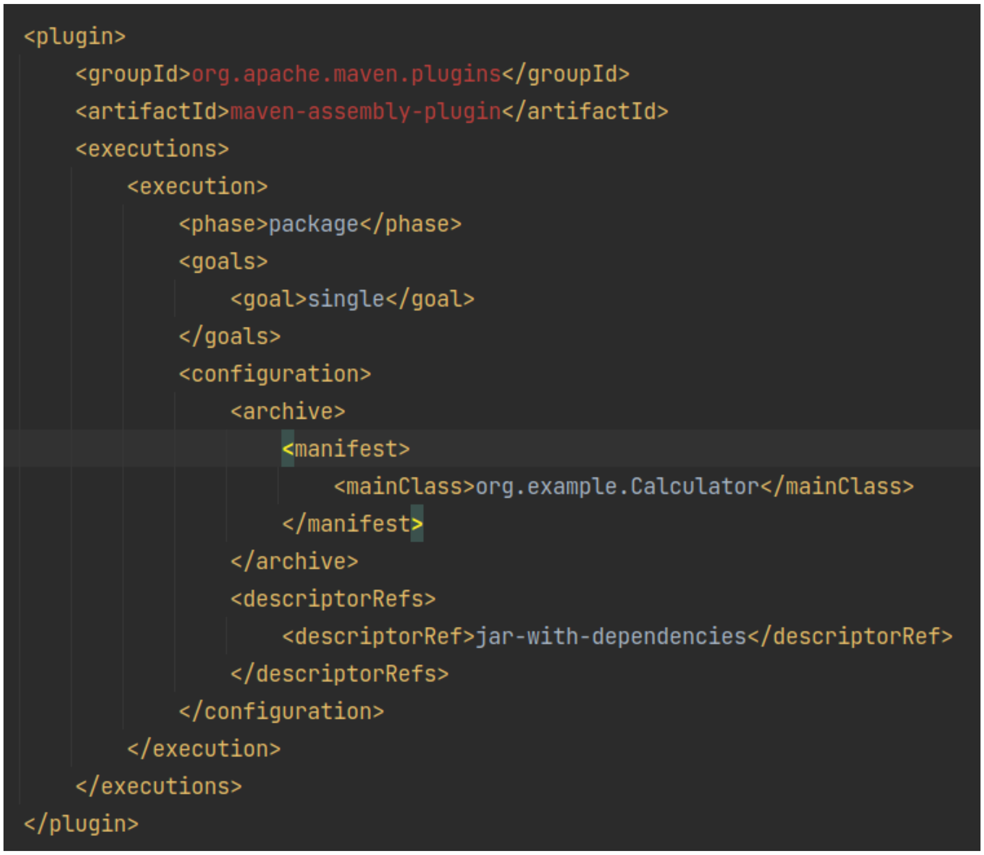


```
no usages   imshukla12
@Test
public void factorialTruePositive(){
    assertEquals( message: "Finding factorial of a number for True Positive",  expected: 720, calculator.fact( num: 6), DELTA);
    assertEquals( message: "Finding factorial of a number for True Positive",  expected: 1, calculator.fact( num: 1), DELTA);
    assertEquals( message: "Finding factorial of a number for True Positive",  expected: 6, calculator.fact( num: 3), DELTA);
    assertEquals( message: "Finding factorial of a number for True Positive",  expected: 24, calculator.fact( num: 4), DELTA);
    assertEquals( message: "Finding factorial of a number for True Positive",  expected: 1, calculator.fact( num: 0), DELTA);
}

no usages   imshukla12
@Test
public void factorialFalsePositive(){
    assertNotEquals( message: "Finding factorial of a number for False Positive",  first: 113, calculator.fact( num: 5), DELTA);
    assertNotEquals( message: "Finding factorial of a number for False Positive",  first: 10, calculator.fact( num: 6), DELTA);
    assertNotEquals( message: "Finding factorial of a number for False Positive",  first: 42, calculator.fact( num: 4), DELTA);
    assertNotEquals( message: "Finding factorial of a number for False Positive",  first: 9, calculator.fact( num: 2), DELTA);
    assertNotEquals( message: "Finding factorial of a number for False Positive",  first: 0, calculator.fact( num: 0), DELTA);
}
```

To use JUnit and log4j, we need to add certain jar files in the pom.xml file. So Maven will add those dependencies.

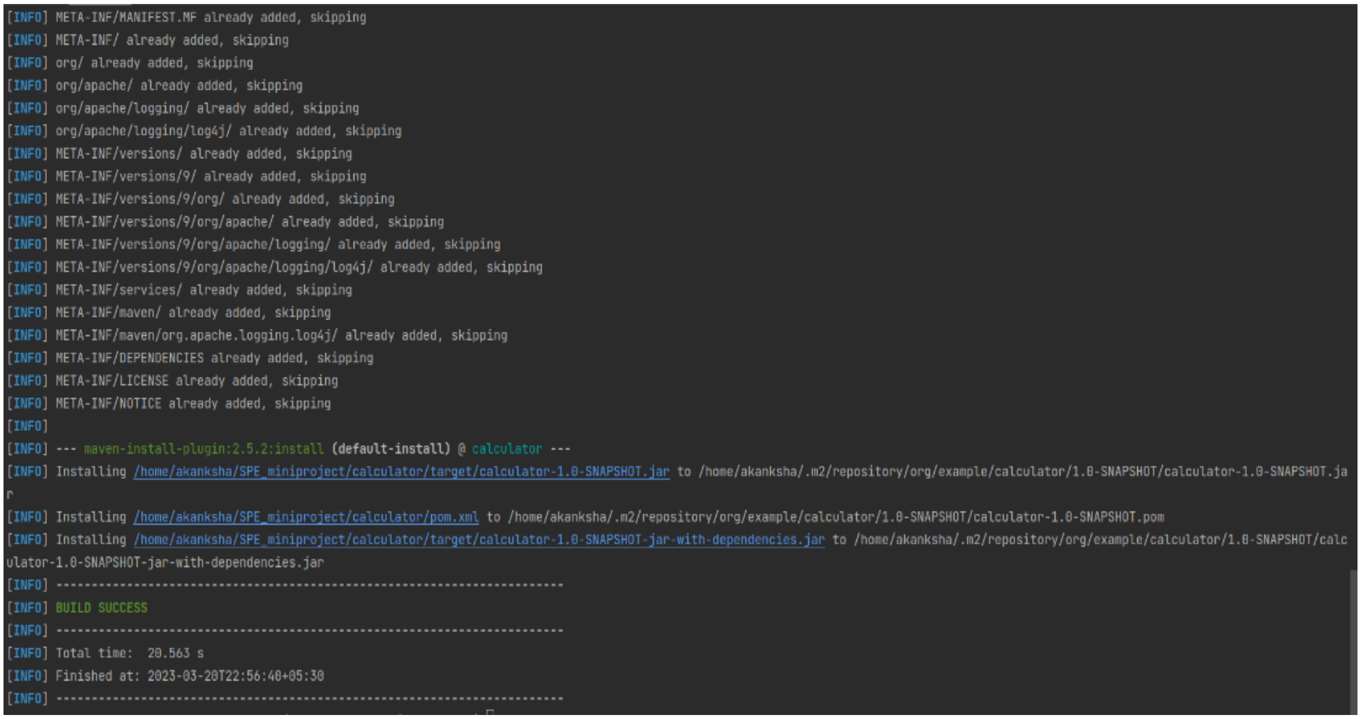


```xml
    <dependencies>
        <dependency>
            <groupId>junit</groupId>
            <artifactId>junit</artifactId>
            <version>4.11</version>
            <scope>test</scope>
        </dependency>

        <dependency>
            <groupId>org.apache.logging.log4j</groupId>
            <artifactId>log4j-api</artifactId>
            <version>2.14.0</version>
        </dependency>
        <dependency>
            <groupId>org.apache.logging.log4j</groupId>
            <artifactId>log4j-core</artifactId>
            <version>2.14.0</version>
        </dependency>

    </dependencies>
```



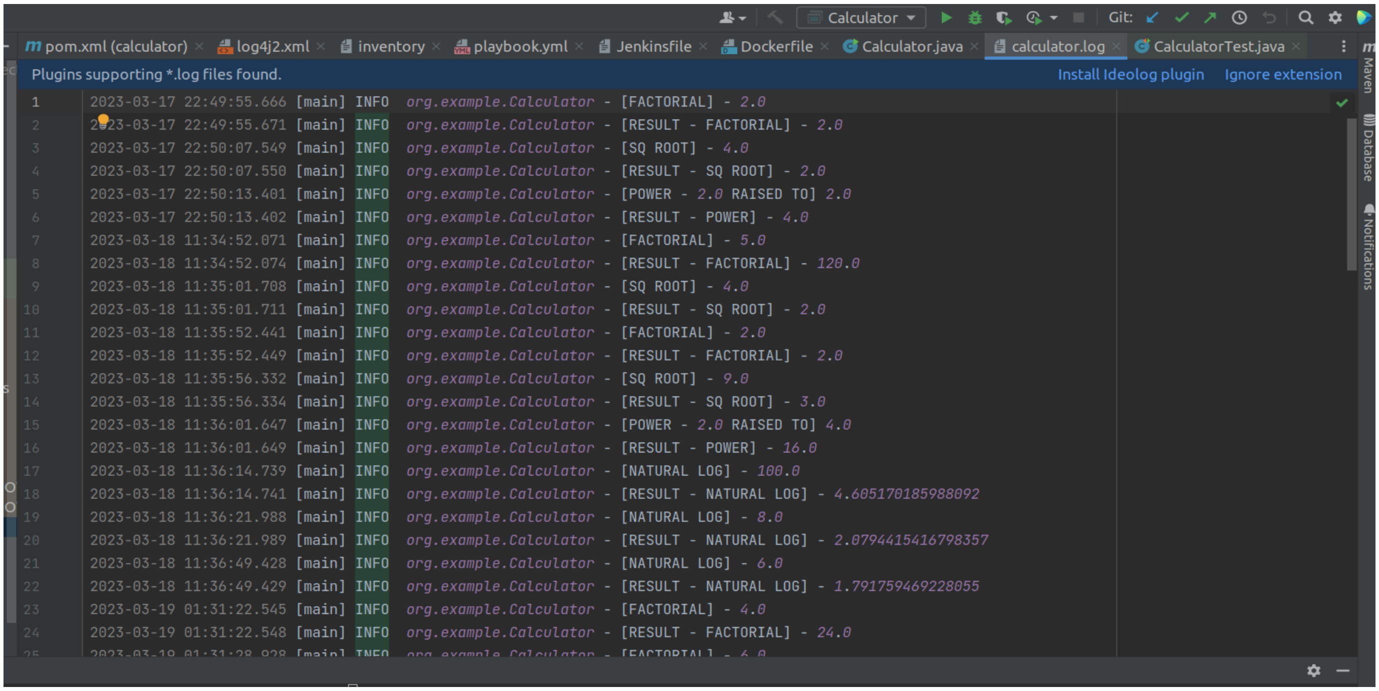
```xml
<plugin>
    <groupId>org.apache.maven.plugins</groupId>
    <artifactId>maven-assembly-plugin</artifactId>
    <executions>
        <execution>
            <phase>package</phase>
            <goals>
                <goal>single</goal>
            </goals>
            <configuration>
                <archive>
                    <manifest>
                        <mainClass>org.example.Calculator</mainClass>
                    </manifest>
                </archive>
                <descriptorRefs>
                    <descriptorRef>jar-with-dependencies</descriptorRef>
                </descriptorRefs>
            </configuration>
        </execution>
    </executions>
</plugin>
```

So now by doing **$ mvn clean install** the complete code will be built and all test cases will be checked and in the current folder, a new folder will get created named "target" in which .jar will be generated.



```
[INFO] META-INF/MANIFEST.MF already added, skipping
[INFO] META-INF/ already added, skipping
[INFO] org/ already added, skipping
[INFO] org/apache/ already added, skipping
[INFO] org/apache/logging/ already added, skipping
[INFO] org/apache/logging/log4j/ already added, skipping
[INFO] META-INF/versions/ already added, skipping
[INFO] META-INF/versions/9/ already added, skipping
[INFO] META-INF/versions/9/org/ already added, skipping
[INFO] META-INF/versions/9/org/apache/ already added, skipping
[INFO] META-INF/versions/9/org/apache/logging/ already added, skipping
[INFO] META-INF/versions/9/org/apache/logging/log4j/ already added, skipping
[INFO] META-INF/services/ already added, skipping
[INFO] META-INF/maven/ already added, skipping
[INFO] META-INF/maven/org.apache.logging.log4j/ already added, skipping
[INFO] META-INF/DEPENDENCIES already added, skipping
[INFO] META-INF/LICENSE already added, skipping
[INFO] META-INF/NOTICE already added, skipping
[INFO]
[INFO] --- maven-install-plugin:2.5.2:install (default-install) @ calculator ---
[INFO] Installing /home/akanksha/SPE_miniproject/calculator/target/calculator-1.0-SNAPSHOT.jar to /home/akanksha/.m2/repository/org/example/calculator/1.0-SNAPSHOT/calculator-1.0-SNAPSHOT.ja
r
[INFO] Installing /home/akanksha/SPE_miniproject/calculator/pom.xml to /home/akanksha/.m2/repository/org/example/calculator/1.0-SNAPSHOT/calculator-1.0-SNAPSHOT.pom
[INFO] Installing /home/akanksha/SPE_miniproject/calculator/target/calculator-1.0-SNAPSHOT-jar-with-dependencies.jar to /home/akanksha/.m2/repository/org/example/calculator/1.0-SNAPSHOT/calc
ulator-1.0-SNAPSHOT-jar-with-dependencies.jar
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  20.563 s
[INFO] Finished at: 2023-03-20T22:56:40+05:30
[INFO] ------------------------------------------------------------------------
```

Running the calculator program will generate calc.log file which looks something like :



Plugins supporting *.log files found. Install Ideolog plugin Ignore extension

```
2023-03-17 22:49:55.666 [main] INFO  org.example.Calculator - [FACTORIAL] - 2.0
2023-03-17 22:49:55.671 [main] INFO  org.example.Calculator - [RESULT - FACTORIAL] - 2.0
2023-03-17 22:50:07.549 [main] INFO  org.example.Calculator - [SQ ROOT] - 4.0
2023-03-17 22:50:07.550 [main] INFO  org.example.Calculator - [RESULT - SQ ROOT] - 2.0
2023-03-17 22:50:13.401 [main] INFO  org.example.Calculator - [POWER - 2.0 RAISED TO] 2.0
2023-03-17 22:50:13.402 [main] INFO  org.example.Calculator - [RESULT - POWER] - 4.0
2023-03-18 11:34:52.071 [main] INFO  org.example.Calculator - [FACTORIAL] - 5.0
2023-03-18 11:34:52.074 [main] INFO  org.example.Calculator - [RESULT - FACTORIAL] - 120.0
2023-03-18 11:35:01.708 [main] INFO  org.example.Calculator - [SQ ROOT] - 4.0
2023-03-18 11:35:01.711 [main] INFO  org.example.Calculator - [RESULT - SQ ROOT] - 2.0
2023-03-18 11:35:52.441 [main] INFO  org.example.Calculator - [FACTORIAL] - 2.0
2023-03-18 11:35:52.449 [main] INFO  org.example.Calculator - [RESULT - FACTORIAL] - 2.0
2023-03-18 11:35:56.332 [main] INFO  org.example.Calculator - [SQ ROOT] - 9.0
2023-03-18 11:35:56.334 [main] INFO  org.example.Calculator - [RESULT - SQ ROOT] - 3.0
2023-03-18 11:36:01.647 [main] INFO  org.example.Calculator - [POWER - 2.0 RAISED TO] 4.0
2023-03-18 11:36:01.649 [main] INFO  org.example.Calculator - [RESULT - POWER] - 16.0
2023-03-18 11:36:14.739 [main] INFO  org.example.Calculator - [NATURAL LOG] - 100.0
2023-03-18 11:36:14.741 [main] INFO  org.example.Calculator - [RESULT - NATURAL LOG] - 4.605170185988092
2023-03-18 11:36:21.988 [main] INFO  org.example.Calculator - [NATURAL LOG] - 8.0
2023-03-18 11:36:21.989 [main] INFO  org.example.Calculator - [RESULT - NATURAL LOG] - 2.0794415416798357
2023-03-18 11:36:49.428 [main] INFO  org.example.Calculator - [NATURAL LOG] - 6.0
2023-03-18 11:36:49.429 [main] INFO  org.example.Calculator - [RESULT - NATURAL LOG] - 1.791759469228055
2023-03-19 01:31:22.545 [main] INFO  org.example.Calculator - [FACTORIAL] - 4.0
2023-03-19 01:31:22.548 [main] INFO  org.example.Calculator - [RESULT - FACTORIAL] - 24.0
2023-03-19 01:31:28.928 [main] INFO  org.example.Calculator - [FACTORIAL] - 6.0
```

Source Code Management – GitHub

The basic goal of SCM is to keep software in its current state (known as the "baseline") while allowing developers to work on new versions for new features or repairs. This is accomplished with the help of GitHub. Create a new repository at https://github.com/ to get started. We can build a new repository by providing it with a unique name connected with the user. The SCM, which will be connected to Jenkins as an input, will manage our code.

Steps:

1. Create a public repository.

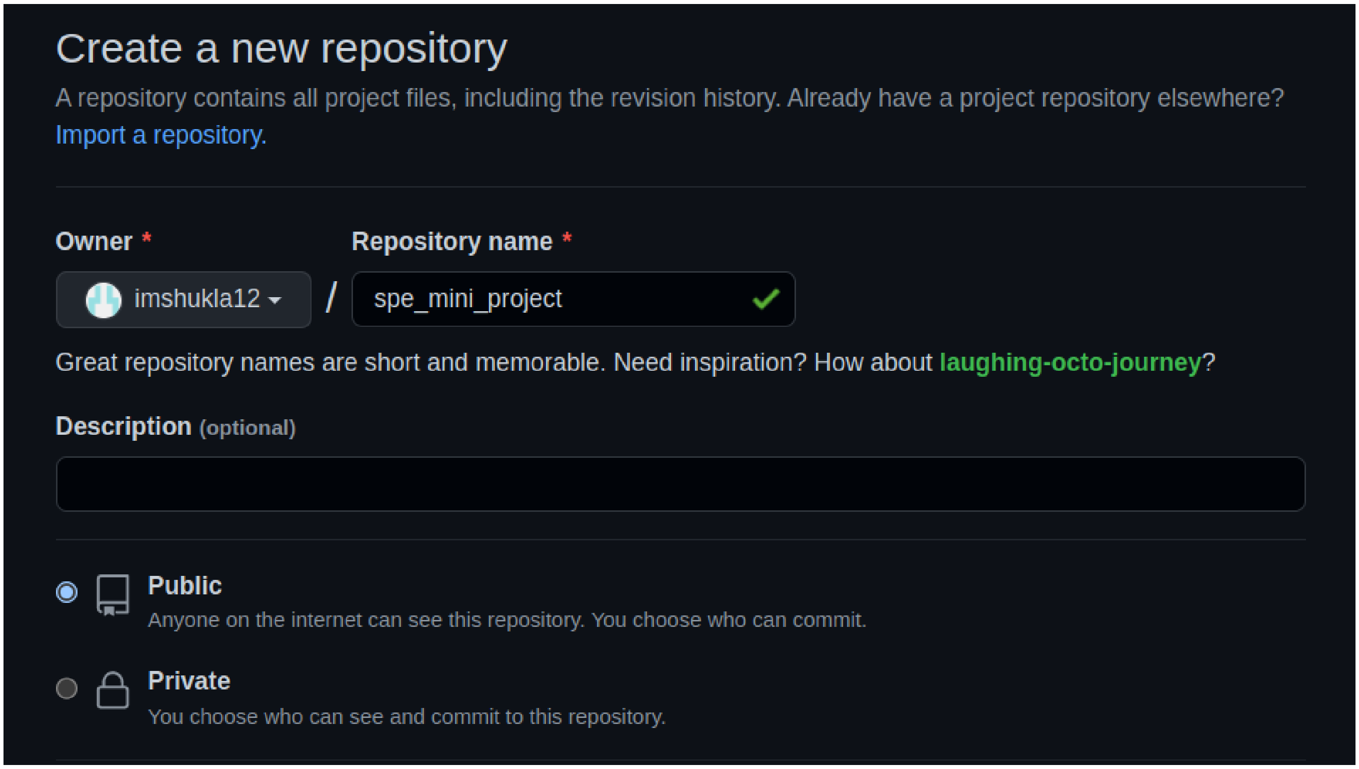
2. $ git init

3. $ git add .

4. $ git remote add origin <github repo URL>

5. $ git commit -m “Message here”

6. $ git push origin master



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

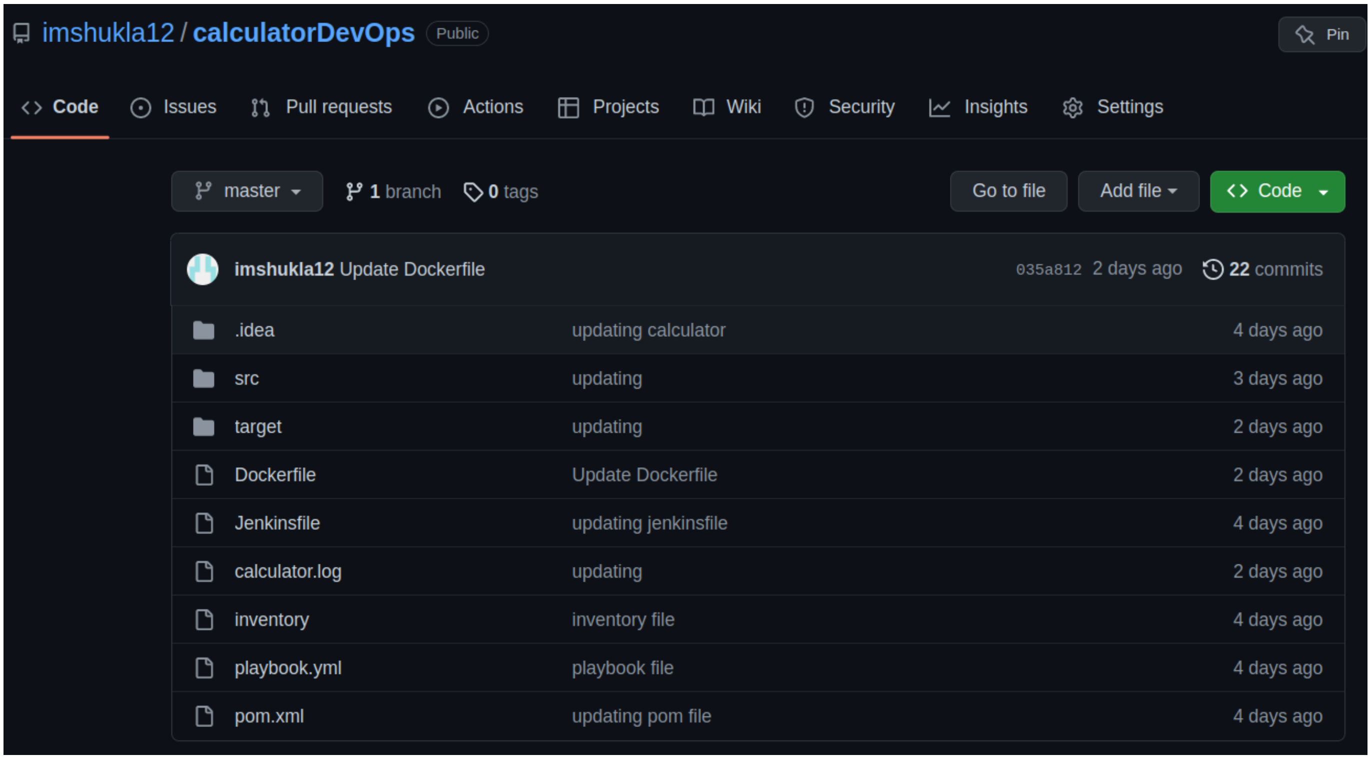
Owner * imshukla12 / Repository name * spe_mini_project ✓

Great repository names are short and memorable. Need inspiration? How about laughing-octo-journey?

Description (optional)

● Public
Anyone on the internet can see this repository. You choose who can commit.

○ Private
You choose who can see and commit to this repository.



imshukla12 Update Dockerfile 035a812 2 days ago 22 commits

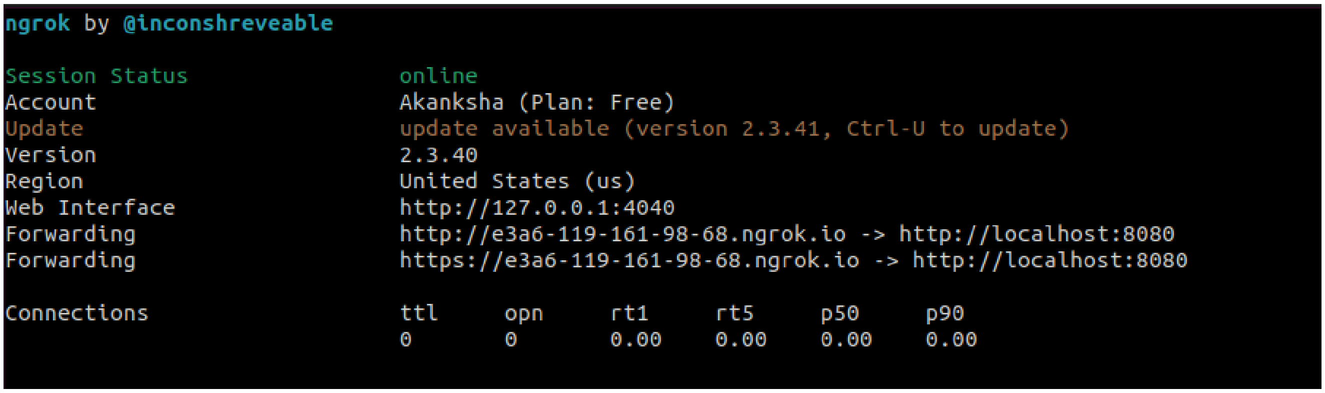
.idea	updating calculator	4 days ago
src	updating	3 days ago
target	updating	2 days ago
Dockerfile	Update Dockerfile	2 days ago
Jenkinsfile	updating jenkinsfile	4 days ago
calculator.log	updating	2 days ago
inventory	inventory file	4 days ago
playbook.yml	playbook file	4 days ago
pom.xml	updating pom file	4 days ago

ngrok

ngrok is a cross-platform application that enables developers to expose a local development server to the Internet with minimal effort.

Setup:

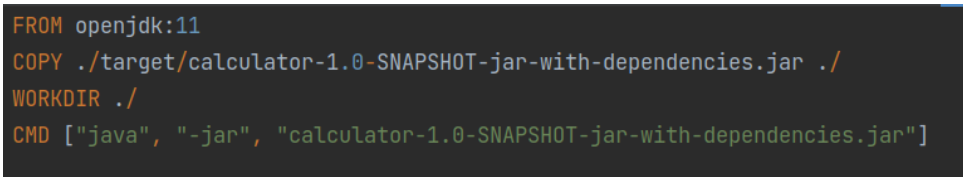
- Go to ngrok website and sign up and then log in to your account.
- Download ngrok for linux
- $unzip /path/to/ngrok.zip
- $ngrok config add-authtoken <your auth token>
- $ngrok http 8080



```
ngrok by @inconshreveable

Session Status                online
Account                       Akanksha (Plan: Free)
Update                        update available (version 2.3.41, Ctrl-U to update)
Version                       2.3.40
Region                        United States (us)
Web Interface                 http://127.0.0.1:4040
Forwarding                    http://e3a6-119-161-98-68.ngrok.io -> http://localhost:8080
Forwarding                    https://e3a6-119-161-98-68.ngrok.io -> http://localhost:8080

Connections                   ttl     opn     rt1     rt5     p50     p90
                              0       0       0.00    0.00    0.00    0.00
```

Docker

Docker is an operating system virtualization platform that allows applications to be delivered in containers. As a result, rather than just supplying software, the full environment is provided as a Docker image, including all software dependencies. So, using open-JDK 11 and the calculator jar file, we'll create a docker image. After that, the image will be posted to the Docker Hub (we need to create a public repository on the docker hub before pushing the image). Ansible will then fetch this image from Docker Hub and deploy it across many machines.



```
FROM openjdk:11
COPY ./target/calculator-1.0-SNAPSHOT-jar-with-dependencies.jar ./
WORKDIR ./
CMD ["java", "-jar", "calculator-1.0-SNAPSHOT-jar-with-dependencies.jar"]
```

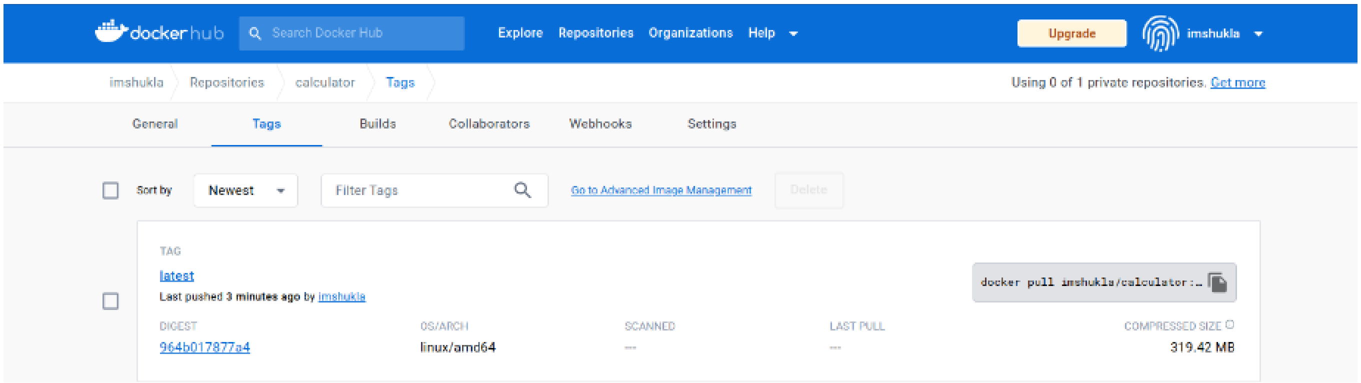
To build a docker image, a docker file is used in which script is written. In the above mentioned docker file,

FROM: It imports the base image openjdk11 inorder to create a new image.

COPY: It can copy a file(should be in the same directory as the Dockerfile) into the image in its root directory.

WORKDIR: it changes the current working directory.

CMD: runs the command inside the image.

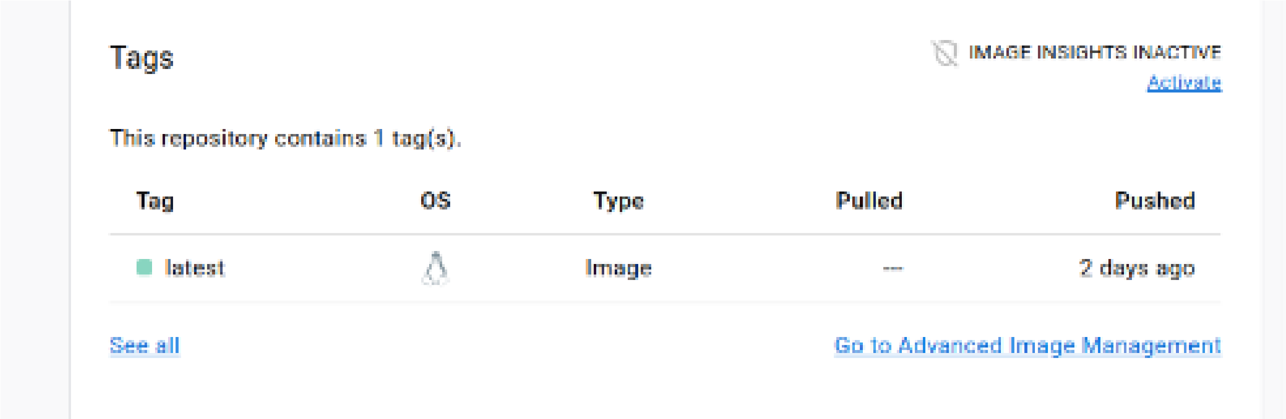


Using 0 of 1 private repositories. Get more

TAG
latest
Last pushed 3 minutes ago by imshukla

```
docker pull imshukla/calculator:…
```

DIGEST	OS/ARCH	SCANNED	LAST PULL	COMPRESSED SIZE
964b017877a4	linux/amd64	---	---	319.42 MB



Tags

This repository contains 1 tag(s).

Tag	OS	Type	Pulled	Pushed
🟩 latest		Image	---	2 days ago

Ansible

Ansible is a configuration management tool that generates written instructions for automating IT professionals' work throughout the entire system infrastructure. Ansible has modules that provide the means of accomplishing a task, the way you use them is through an Ansible playbook. A playbook is a configuration file written in YAML that provides instructions for what needs to be done in order to bring a managed node into the desired state.

We will use SSH for establishing communication between Ansible Engine and Ansible Node. As the SSH protocol is widely used for communication in cloud services, network environments, file transfer tools, configuration management tools, and other computer-dependent services, to authenticate identity and protect those services from unintended use or malicious attacks.Steps to Install :

- sudo apt install openssh-server
- ssh-keygen -t rsa

- ssh-copy-id <username>@<IP>
- sudo apt install ansible

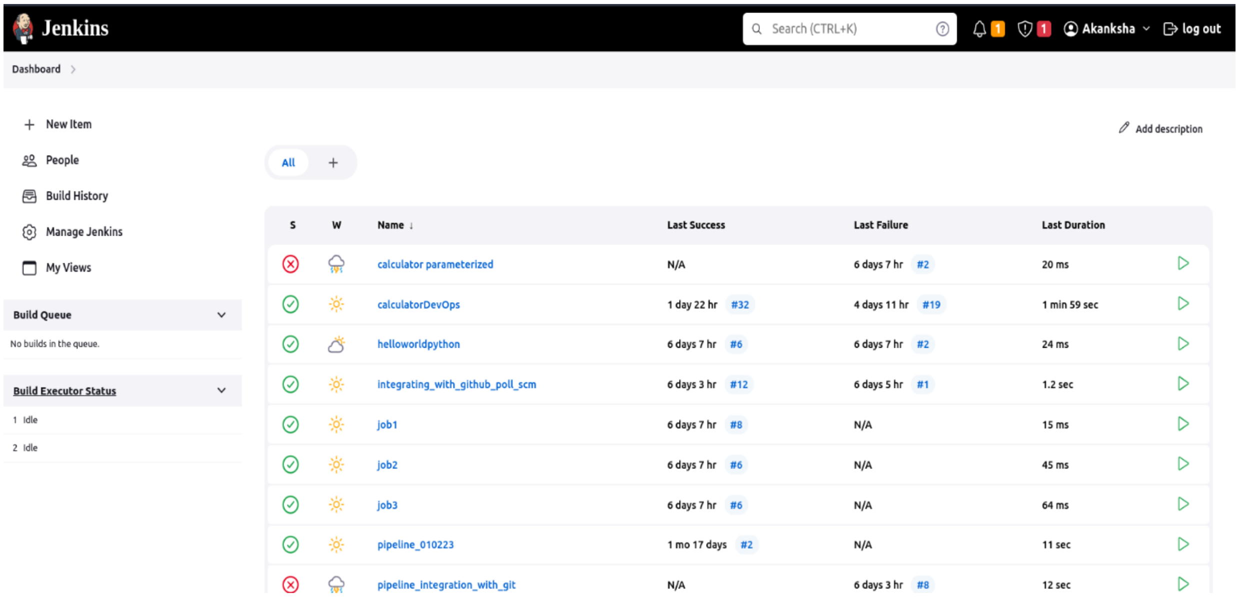
Jenkins

Jenkins is an open-source automation tool written in Java with plugins built for continuous integration. Jenkins is used to building and test your software projects continuously making it easier for developers to integrate changes to the project, and making it easier for users to obtain a fresh build. It also allows you to continuously deliver your software by integrating with a large number of testing and deployment technologies.

Setup Jenkins on your local machine:

- $ wget -q -O - https://pkg.jenkins.io/debian-stable/jenkins.io.key |sudo apt-key add –
- $ sudo sh -c 'echo deb http://pkg.jenkins.io/debian-stable
- $ /etc/apt/sources.list.d/jenkins.list'
- $ sudo apt update
- $ sudo apt install Jenkins

The Jenkins pipeline was utilized in this project to handle until delivery, i.e. continuous delivery. **http://localhost:8080** is the URL for the Jenkins service. To go to it, open a web browser and type this URL into the address bar.



No builds in the queue.

1 Idle

2 Idle

S	W	Name ↓	Last Success	Last Failure	Last Duration	
✗	⛈	calculator parameterized	N/A	6 days 7 hr #2	20 ms	▷
✓	☀	calculatorDevOps	1 day 22 hr #32	4 days 11 hr #19	1 min 59 sec	▷
✓	⛅	helloworldpython	6 days 7 hr #6	6 days 7 hr #2	24 ms	▷
✓	☀	integrating_with_github_poll_scm	6 days 3 hr #12	6 days 5 hr #1	1.2 sec	▷
✓	☀	job1	6 days 7 hr #8	N/A	15 ms	▷
✓	☀	job2	6 days 7 hr #6	N/A	45 ms	▷
✓	☀	job3	6 days 7 hr #6	N/A	64 ms	▷
✓	☀	pipeline_010223	1 mo 17 days #2	N/A	11 sec	▷
✗	⛈	pipeline_integration_with_git	N/A	6 days 3 hr #8	12 sec	▷

Prerequisites to getting started to

1. Install the following plugins from the plugin manager

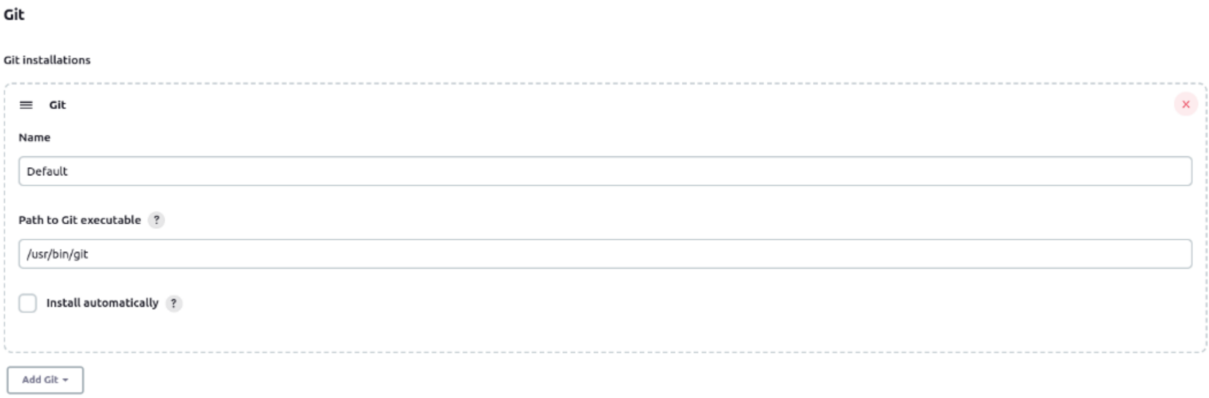
- Maven
- Git
- Ansible
- Docker

Go to manage Jenkins -> Manage Plugins -> Search for these plugins in the available plugins. And then check if installed under Installed Plugins.

2. Before going to the next step, make sure that the following are installed in your local machine by using <packageName> –version. If not installed, then do

- Git -> $sudo apt install git
- maven-> maven installation
- Ansible -> $sudo apt install ansible
- Docker -> $sudo apt install docker.io

3. Manage Jenkins -> Global Tool Configuration



Git

Git installations

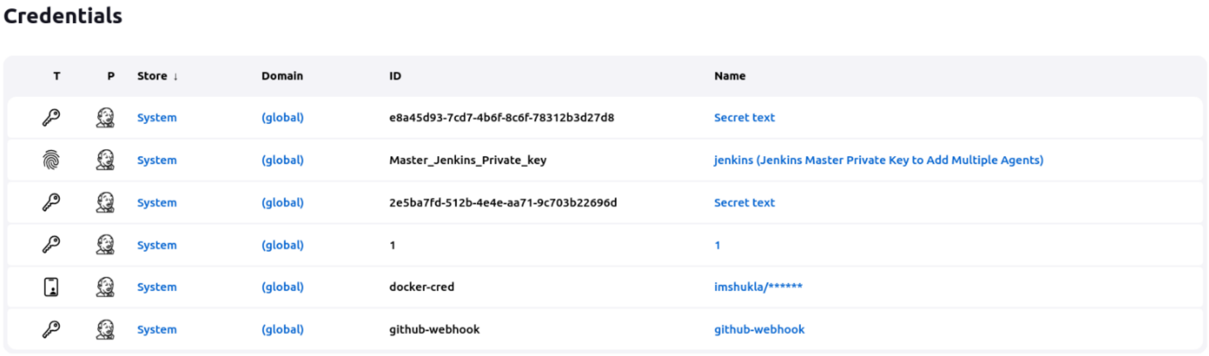
Git

Name: Default

Path to Git executable: /usr/bin/git

☐ Install automatically

4. Manage Jenkins ⟶ Manage Credentials

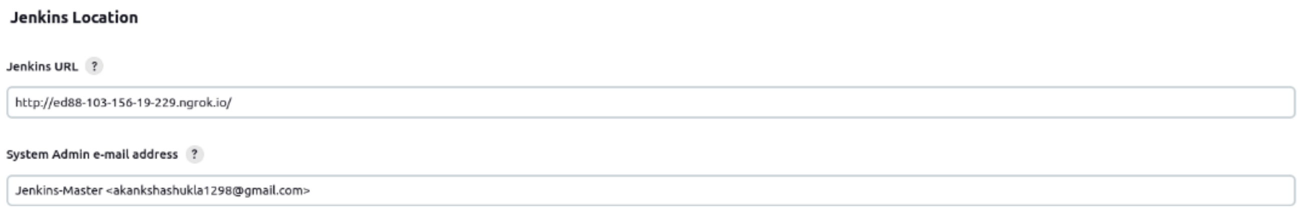


Credentials

T	P	Store ↓	Domain	ID	Name
		System	(global)	e8a45d93-7cd7-4b6f-8c6f-78312b3d27d8	Secret text
		System	(global)	Master_Jenkins_Private_key	jenkins (Jenkins Master Private Key to Add Multiple Agents)
		System	(global)	2e5ba7fd-512b-4e4e-aa71-9c703b22696d	Secret text
		System	(global)	1	1
		System	(global)	docker-cred	imshukla/******
		System	(global)	github-webhook	github-webhook

5. Manage Jenkins ⟶ Configure System

Using ngrok expose your localhost to the internet $ngrok http 8080 and copy the forwarding url, then paste it in Jenkins URL.



Jenkins Location

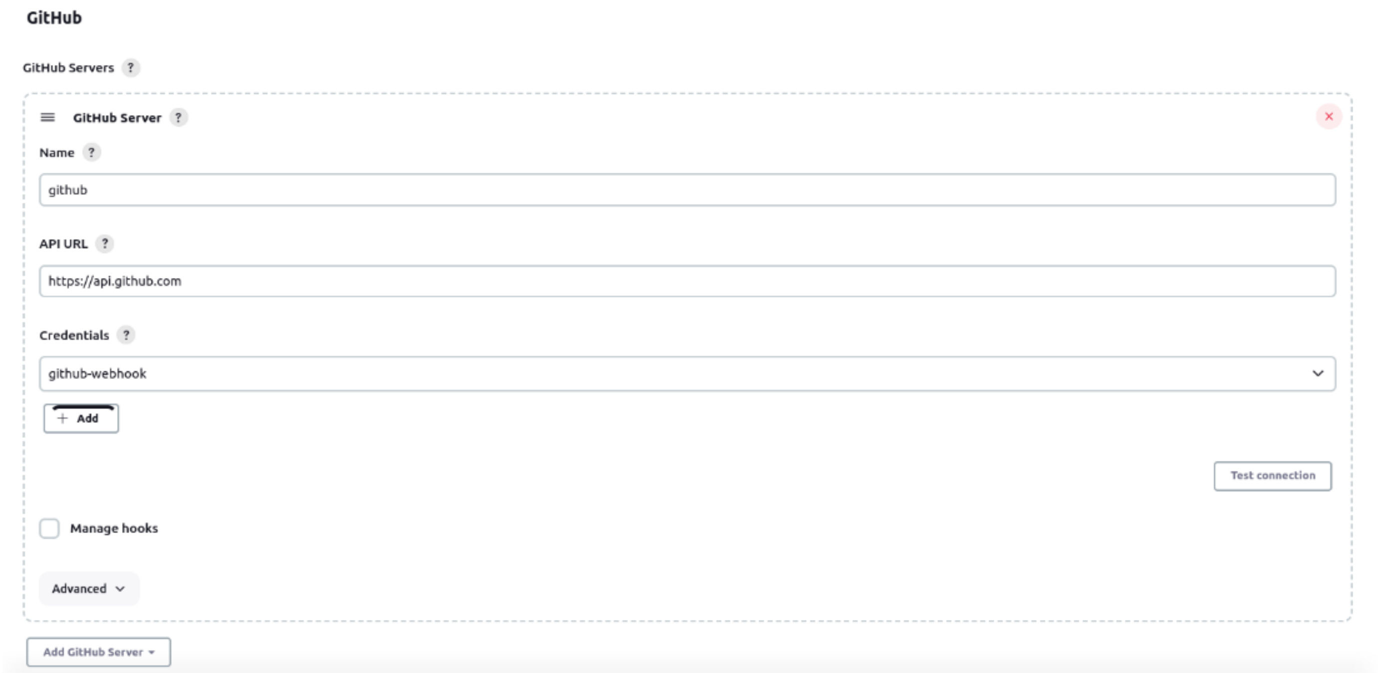
Jenkins URL ?

http://ed88-103-156-19-229.ngrok.io/

System Admin e-mail address ?

Jenkins-Master <akankshashukla1298@gmail.com>

Under the GitHub server add the following. Then click apply and save.



GitHub

GitHub Servers ?

GitHub Server ?

Name ?

github

API URL ?

https://api.github.com

Credentials ?

github-webhook

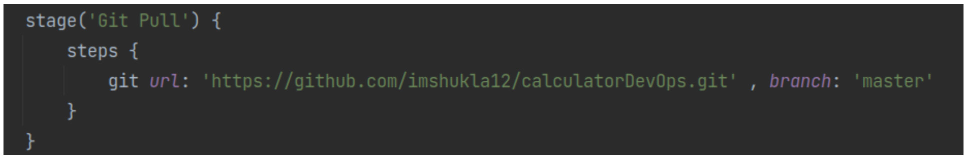
□ Manage hooks

Jenkins Pipeline

Sequence of stages to perform the given tasks such as pulling code from the Git repository, static code analysis, building project, executing the unit tests, automated tests, performance tests, and deploying application.

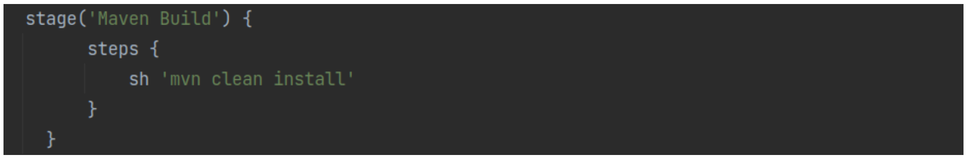
Steps :

1. Git Pull: It pulls the remote repository from GitHub using Jenkins.



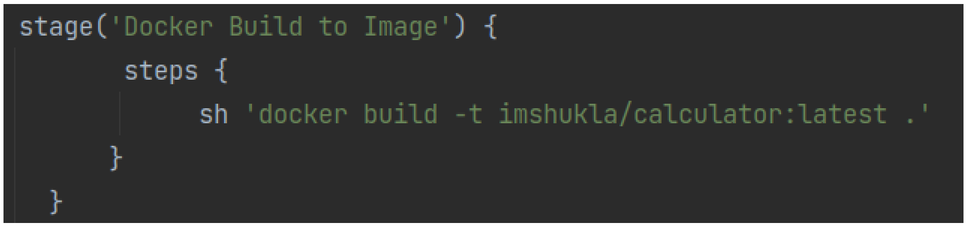
```
stage('Git Pull') {
    steps {
        git url: 'https://github.com/imshukla12/calculatorDevOps.git' , branch: 'master'
    }
}
```

2. Maven Build: It generates a jar file that contains our source code as well as any dependencies. The existing target folder with old dependencies will be deleted, and a new target folder with the new jar file will be created.



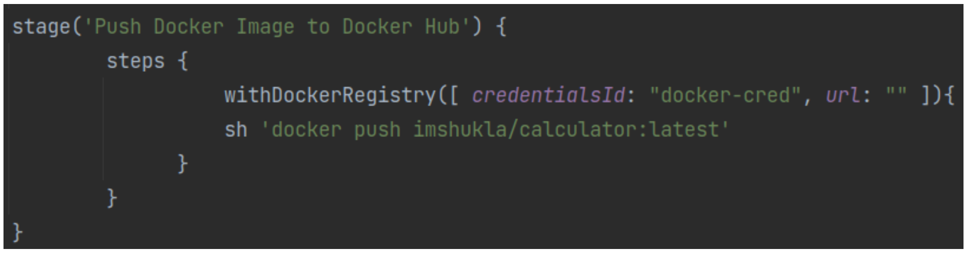
```
stage('Maven Build') {
      steps {
          sh 'mvn clean install'
      }
  }
```

3. Docker Image Creation: It's used to produce images on our local system that are then posted to our Docker hub, allowing us to pull the image and run the application on other servers. environment just creates variables which can be used later. :latest is the tag name of the image.



```
stage('Docker Build to Image') {
       steps {
            sh 'docker build -t imshukla/calculator:latest .'
      }
  }
```

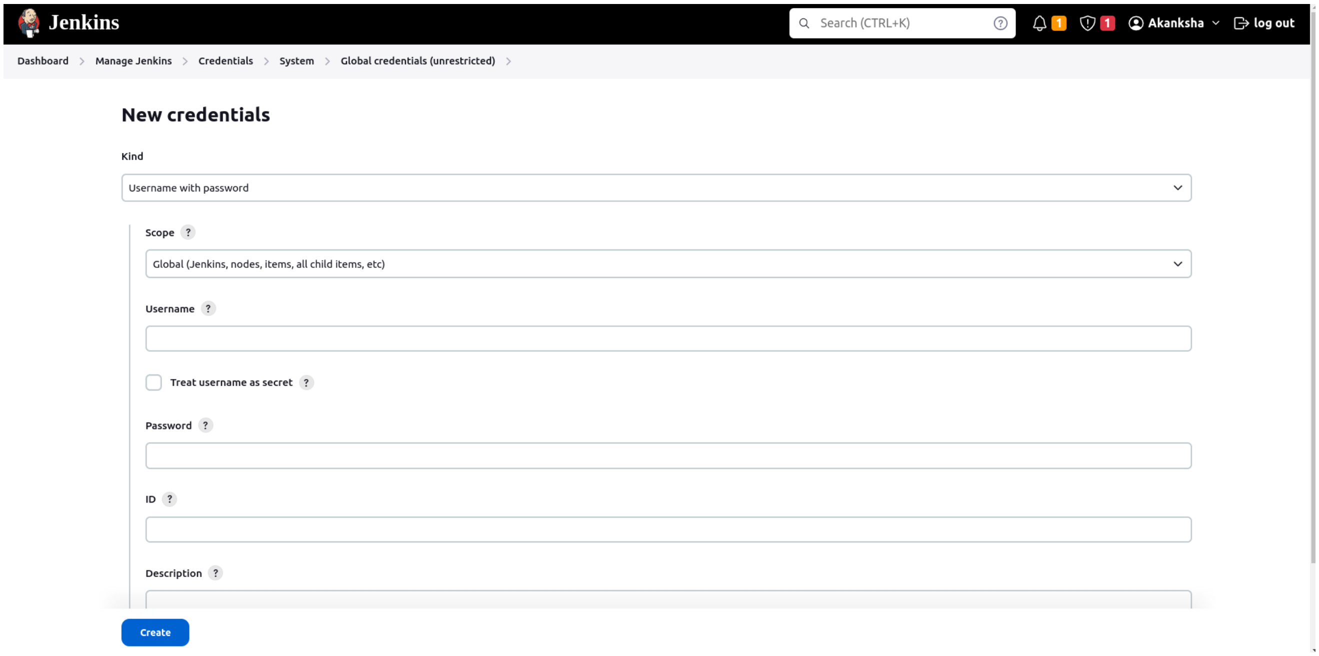
4. Deploying Docker Image: Here we are deploying the image into Docker Hub so that anyone can pull the image. We have to run this command $ sudo chmod 666 /var/run/docker.sock in localhost in order to give the permission.



```
stage('Push Docker Image to Docker Hub') {
        steps {
                withDockerRegistry([ credentialsId: "docker-cred", url: "" ]){
                sh 'docker push imshukla/calculator:latest'
            }
        }
}
```

To use the **withCredentials** function, we have to save our credentials in the Jenkins which can be done as follows:

1. Manage Jenkins -> Under security click on Manage Credentials -> click on global -> click on Add Credentials.
2. Select secret text under ‘kind’ -> select scope -> add your Docker Hub password as secret -> give it an id -> add a description if you want -> click on Create.


Jenkins
Search (CTRL+K)
1
1
Akanksha
log out
Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted) >
Create

New credentials

Kind: Username with password

Scope: Global (Jenkins, nodes, items, all child items, etc)

Username:

□ Treat username as secret

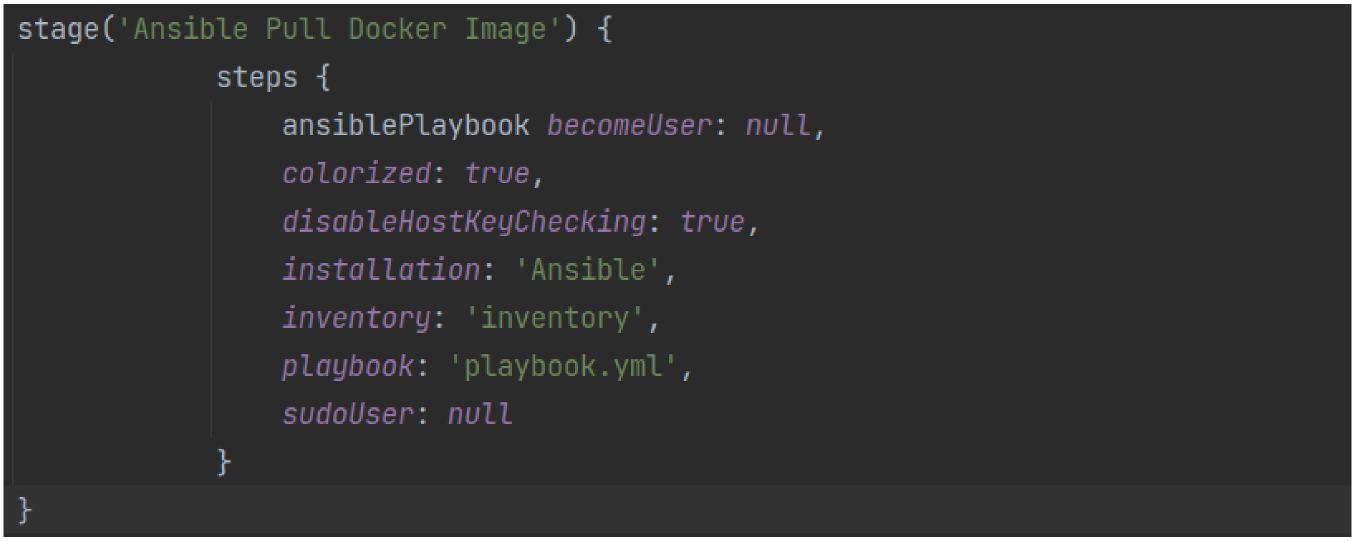
Password:

ID:

Description:

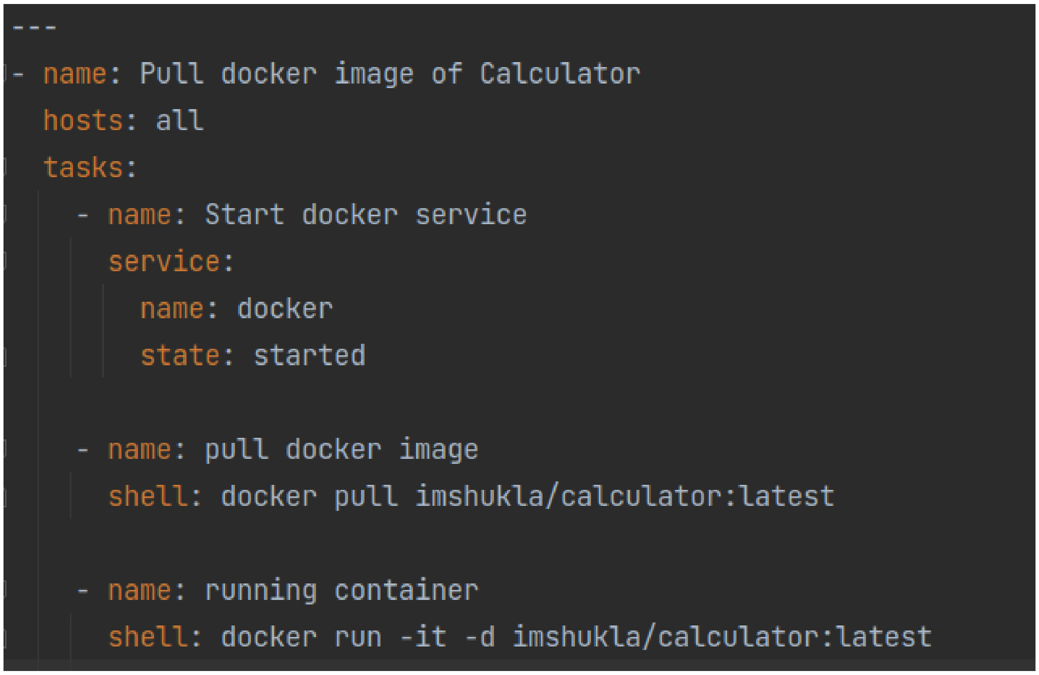
5. Ansible Deploy:

In this stage, we run our ansible playbook. The ansible playbook contains all the commands to be executed on a machine (called ansible node) and the ansible_node details are mentioned in the inventory file.



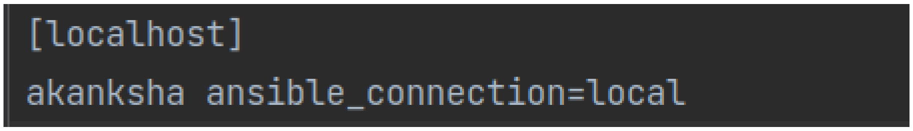
```
stage('Ansible Pull Docker Image') {
            steps {
                ansiblePlaybook becomeUser: null,
                colorized: true,
                disableHostKeyChecking: true,
                installation: 'Ansible',
                inventory: 'inventory',
                playbook: 'playbook.yml',
                sudoUser: null
            }
}
```

- The Playbook file looks like this:



```
---
- name: Pull docker image of Calculator
  hosts: all
  tasks:
    - name: Start docker service
      service:
        name: docker
        state: started

    - name: pull docker image
      shell: docker pull imshukla/calculator:latest

    - name: running container
      shell: docker run -it -d imshukla/calculator:latest
```

- And the inventory file should look something like this:



```
[localhost]
akanksha ansible_connection=local
```

Automatically Trigger the Pipeline using Webhooks :

Webhooks are messages that are sent automatically whenever something changes. In our scenario, the webhook will automatically start the Jenkins pipeline if we make any updates to the GitHub repo.

Ngrok uses secure tunnels to connect local servers behind NATs (Network Address Translation) and firewalls to the public internet. It has a real-time web interface that allows us to inspect any HTTP traffic passing through your tunnels. It allows you to connect to the internet via a web server running on your local system.

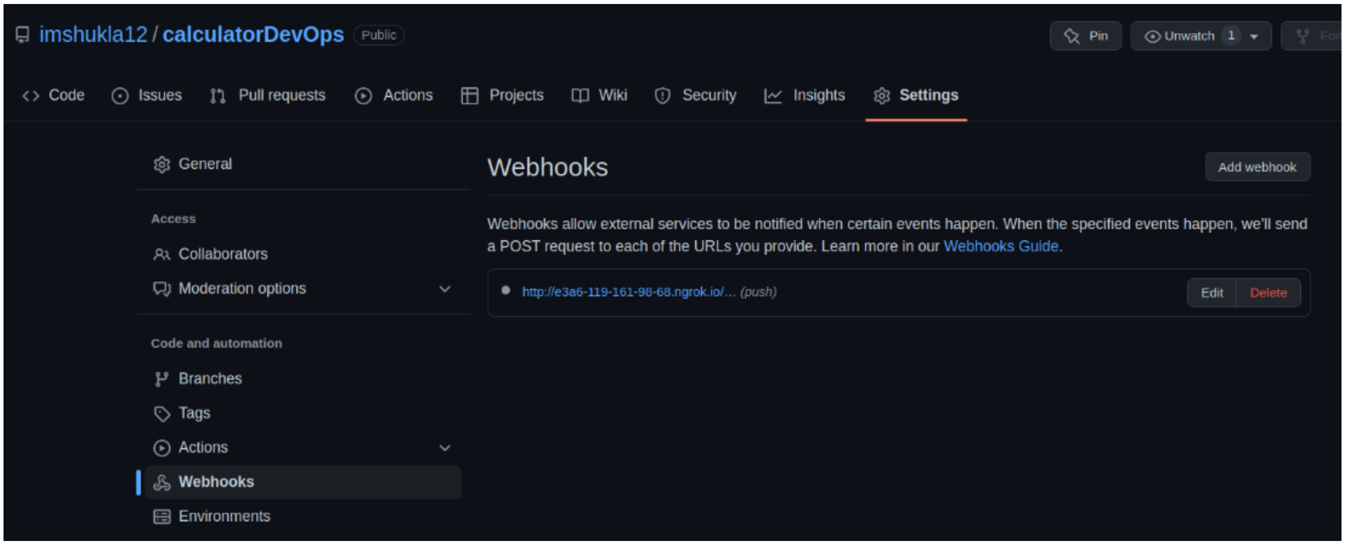
Webhooks allow us to build or set up integrations, such as GitHub Apps, which subscribe to certain events on GitHub.com. When one of those events is triggered, we'll send an HTTP POST payload to the webhook's configured URL.

Setup webhook

- Go to your github account -> settings -> on the left pane, click on Developer Settings -> Click on personal access token -> select classic token -> click on generate new token (classic)

- Give a token name and check admin:repo_hook -> click on Generate Token. Make sure to copy the personal access token.
- Now go to your project repository -> click on repository settings -> select webhooks on the left pane -> click on Add Webhook
- Paste the ngrok url: $ngrok http 8080 (copy the forwarding URL)

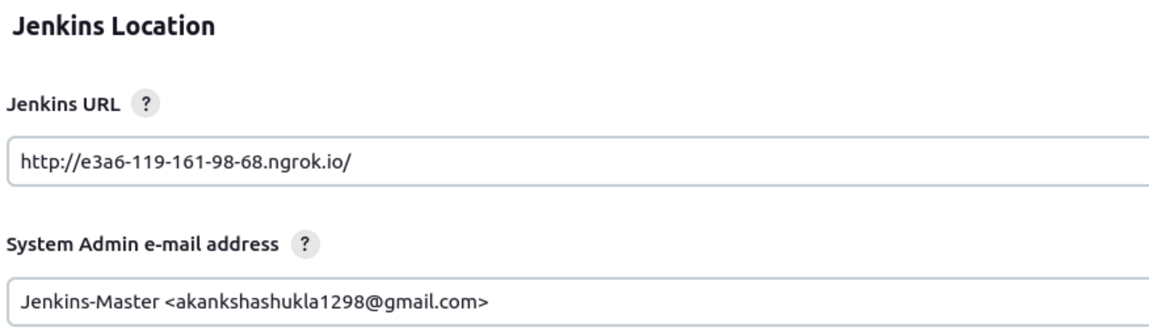
NOTE - ngrok url changes every time you start the service so make sure to change the url wherever you copied it in the entire project i.e., Github webhook and Jenkins url in Jenkins' Configure System.



Webhooks

Webhooks allow external services to be notified when certain events happen. When the specified events happen, we'll send a POST request to each of the URLs you provide. Learn more in our Webhooks Guide.

http://e3a6-119-161-98-68.ngrok.io/... (push)



Jenkins Location

Jenkins URL ?

http://e3a6-119-161-98-68.ngrok.io/

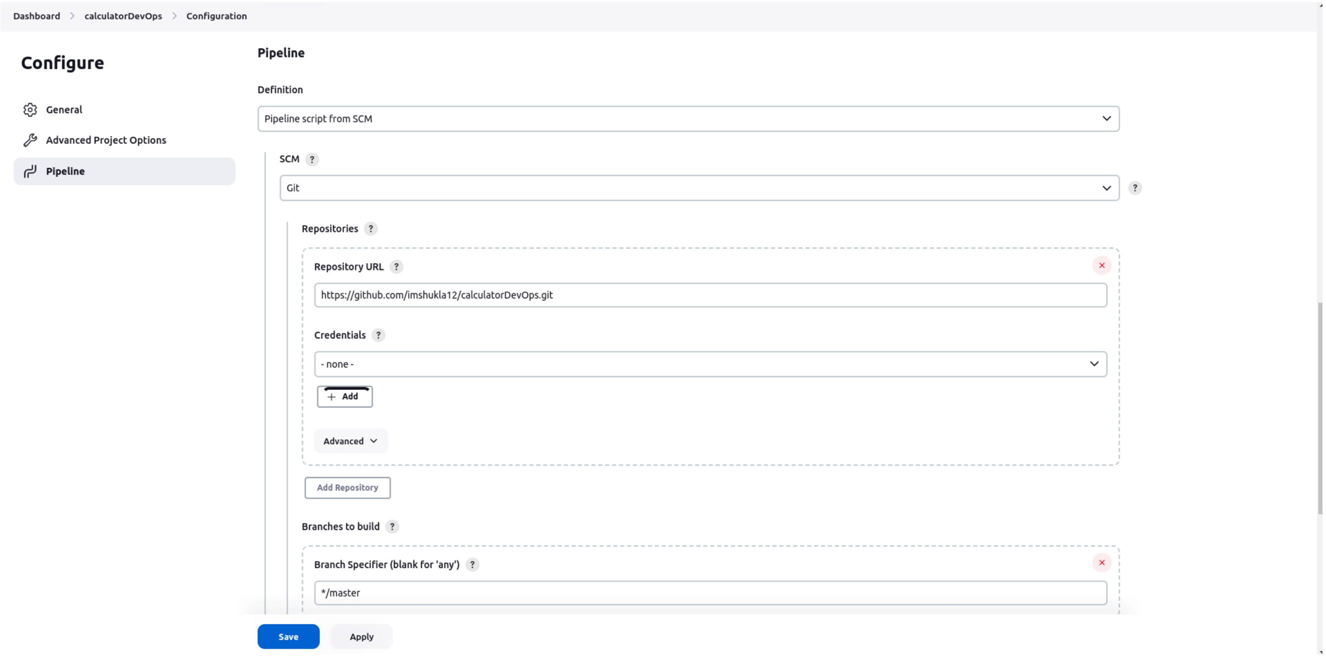
System Admin e-mail address ?

Jenkins-Master <akankshashukla1298@gmail.com>

Setup Jenkins project

- Select the project -> click on configure -> under Build Triggers, check the GitHub hook trigger for GitSCM polling
- Now copy the entire pipeline script -> Make a file named 'Jenkinsfile' in the project directory and paste the pipeline script in it.

- Now under project configuration -> select pipeline script for SCM
    a. Select git as SCM.
    b. Paste the GitHub repo URL.
    c. Select branch.
    d. Give Jenkinsfile path.


Dashboard > calculatorDevOps > Configuration

Configure

Pipeline

Definition: Pipeline script from SCM

SCM: Git

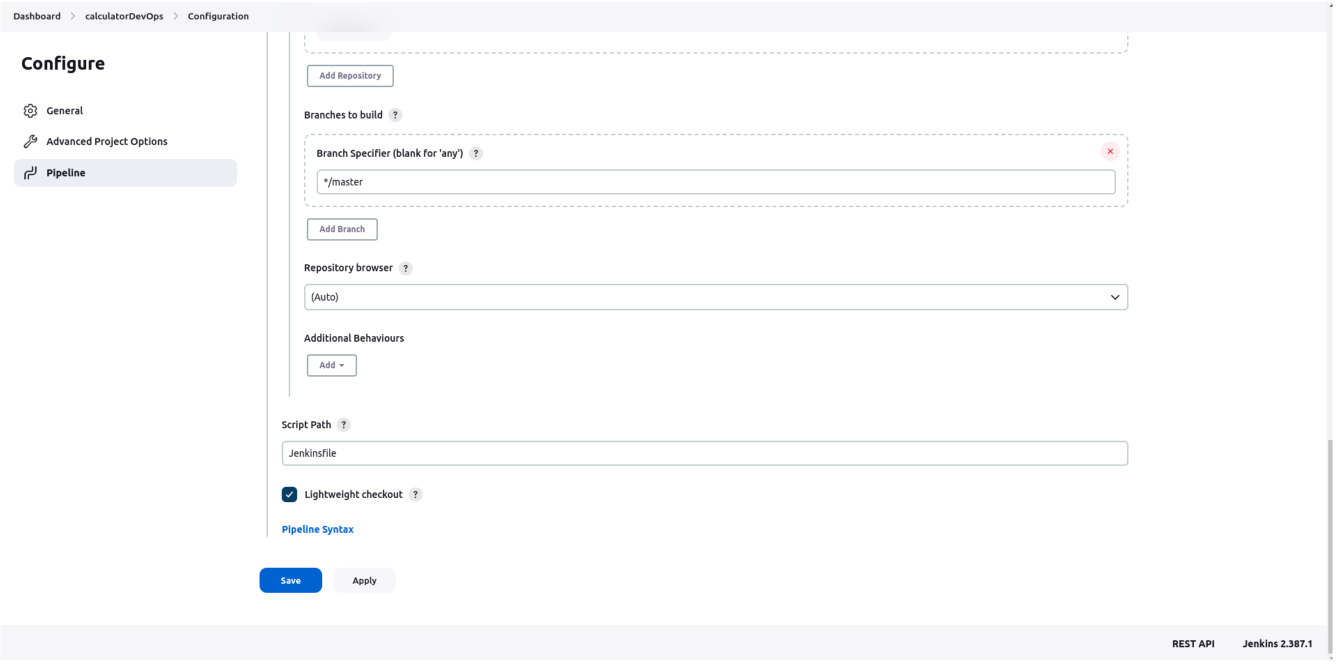
Repositories

Repository URL: https://github.com/imshukla12/calculatorDevOps.git

Credentials: - none -

Branches to build

Branch Specifier (blank for 'any'): */master


Dashboard > calculatorDevOps > Configuration

Configure

Branches to build

Branch Specifier (blank for 'any'): */master

Repository browser: (Auto)

Additional Behaviours

Script Path: Jenkinsfile

☑ Lightweight checkout

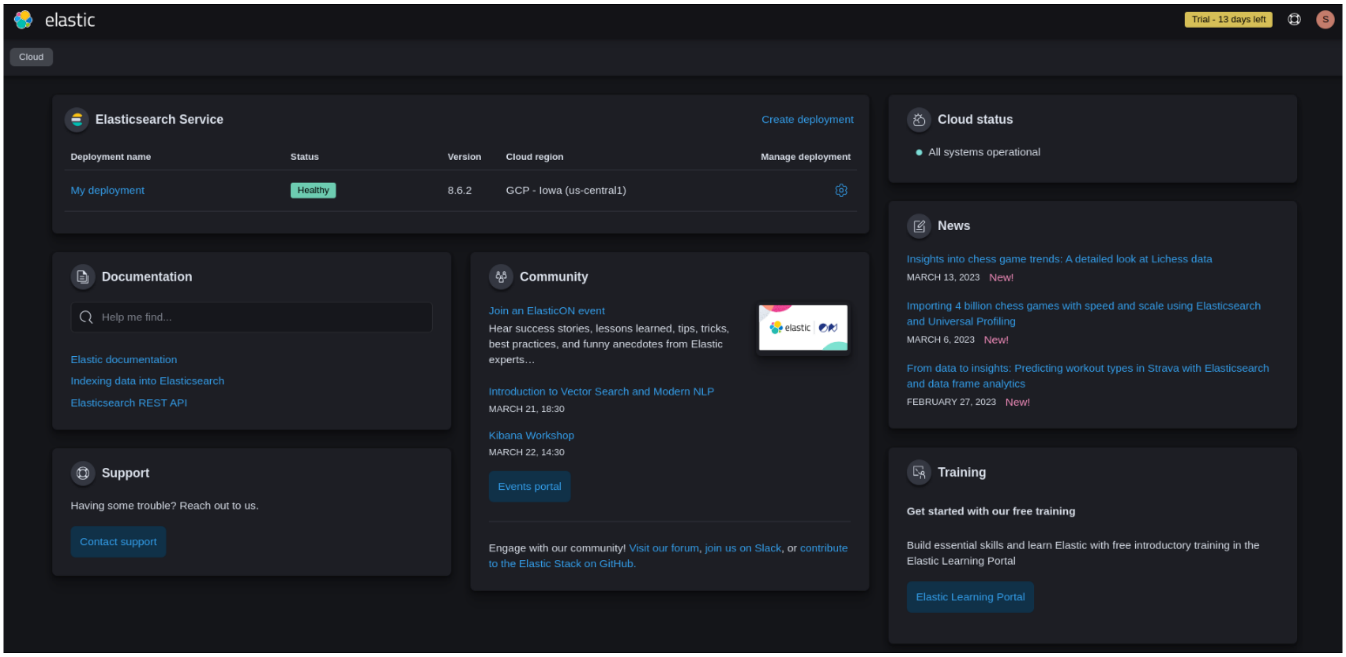
ELK Stack

After you've completed your deployment, the following step is to monitor your system. Monitoring entails determining whether or not the software is performing as intended.

ELK stack makes the monitoring tool for any deployed software, it analyzes the logs and the same analysis can then be viewed on the Kibana dashboard. ELK stack is comprised of 3 independent components: Elasticsearch, Logstash, Kibana.

Steps :

- Go to the Elastic Cloud website -> signup using a email-id.
- Under get started by adding integrations -> click on file upload -> upload the calc.log file which we have generated in our program.
- Now click on override settings -> select data format as “semi_structured_text”.
- For Grok pattern, go to Grok constructor website and paste your pattern which you have written in log4j2.xml file.
- After you apply, click on Import (in bottom left).
- Give an index name and click import.
- Click on view index and discover.



Elasticsearch Service

Deployment name	Status	Version	Cloud region	Manage deployment
My deployment	Healthy	8.6.2	GCP - Iowa (us-central1)	⚙

Cloud status

- All systems operational

Documentation

Elastic documentation
Indexing data into Elasticsearch
Elasticsearch REST API

Community

Join an ElasticON event
Hear success stories, lessons learned, tips, tricks, best practices, and funny anecdotes from Elastic experts…

Introduction to Vector Search and Modern NLP
MARCH 21, 18:30

Kibana Workshop
MARCH 22, 14:30

Engage with our community! Visit our forum, join us on Slack, or contribute to the Elastic Stack on GitHub.

News

Insights into chess game trends: A detailed look at Lichess data
MARCH 13, 2023 New!

Importing 4 billion chess games with speed and scale using Elasticsearch and Universal Profiling
MARCH 6, 2023 New!

From data to insights: Predicting workout types in Strava with Elasticsearch and data frame analytics
FEBRUARY 27, 2023 New!

Support

Having some trouble? Reach out to us.

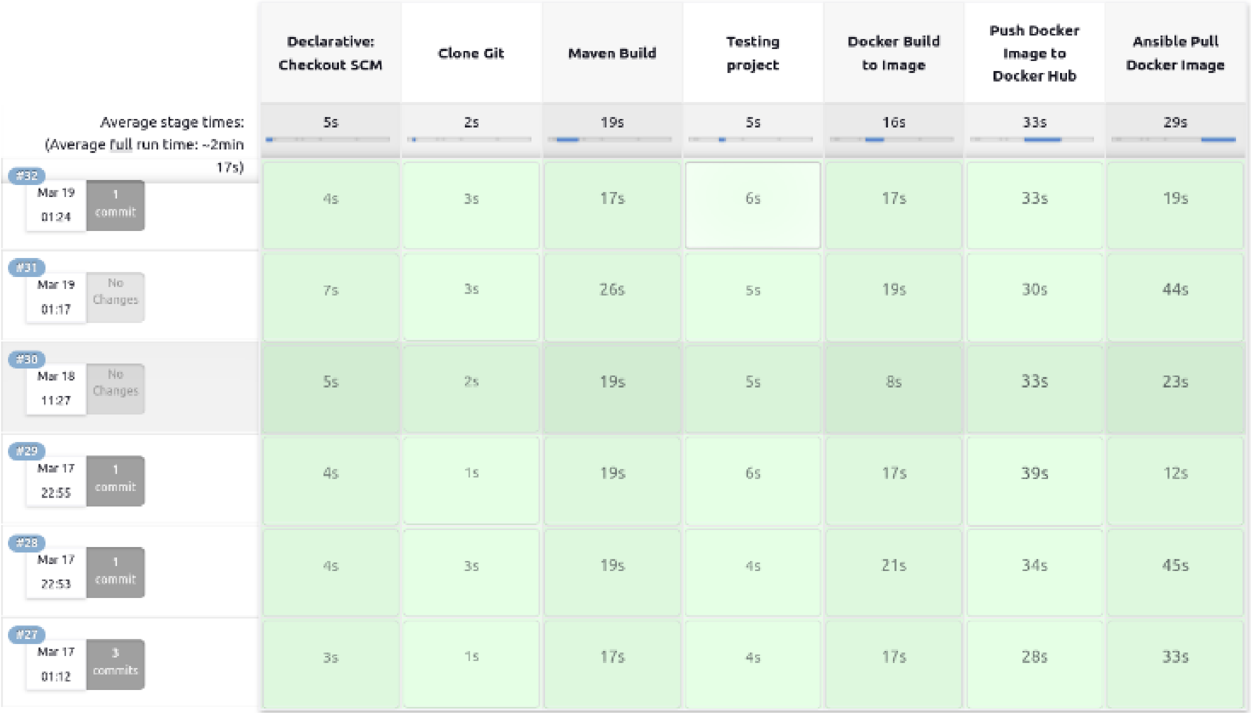
Training

Get started with our free training

Build essential skills and learn Elastic with free introductory training in the Elastic Learning Portal

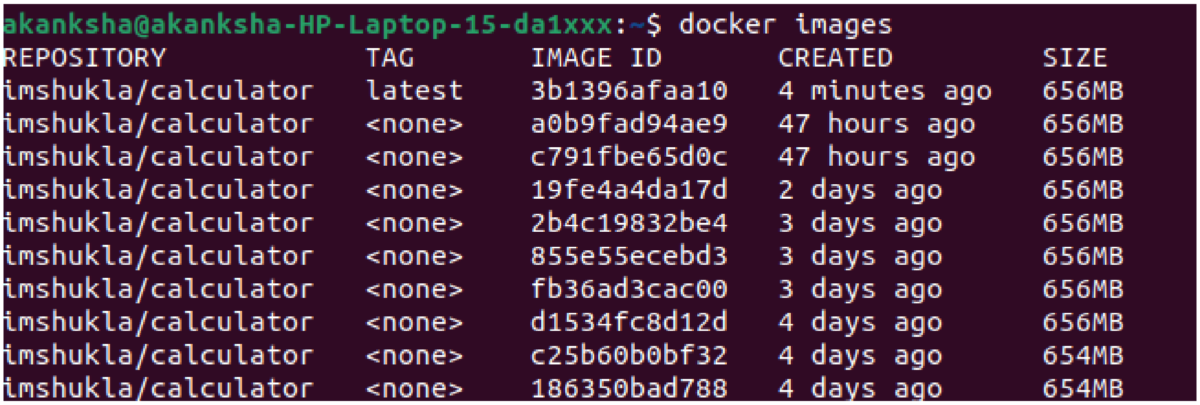
Output

Output of the Jenkins pipeline project

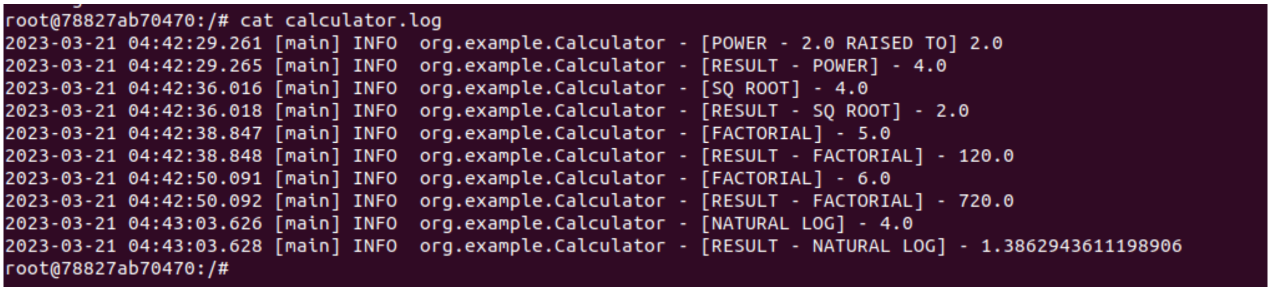


	Declarative: Checkout SCM	Clone Git	Maven Build	Testing project	Docker Build to Image	Push Docker Image to Docker Hub	Ansible Pull Docker Image
Average stage times: (Average full run time: ~2min 17s)	5s	2s	19s	5s	16s	33s	29s
#32 Mar 19 01:24 1 commit	4s	3s	17s	6s	17s	33s	19s
#31 Mar 19 01:17 No Changes	7s	3s	26s	5s	19s	30s	44s
#30 Mar 18 11:27 No Changes	5s	2s	19s	5s	8s	33s	23s
#29 Mar 17 22:55 1 commit	4s	1s	19s	6s	17s	39s	12s
#28 Mar 17 22:53 1 commit	4s	3s	19s	4s	21s	34s	45s
#27 Mar 17 01:12 3 commits	3s	1s	17s	4s	17s	28s	33s

Output in the local machine: New repository of our docker image is added.

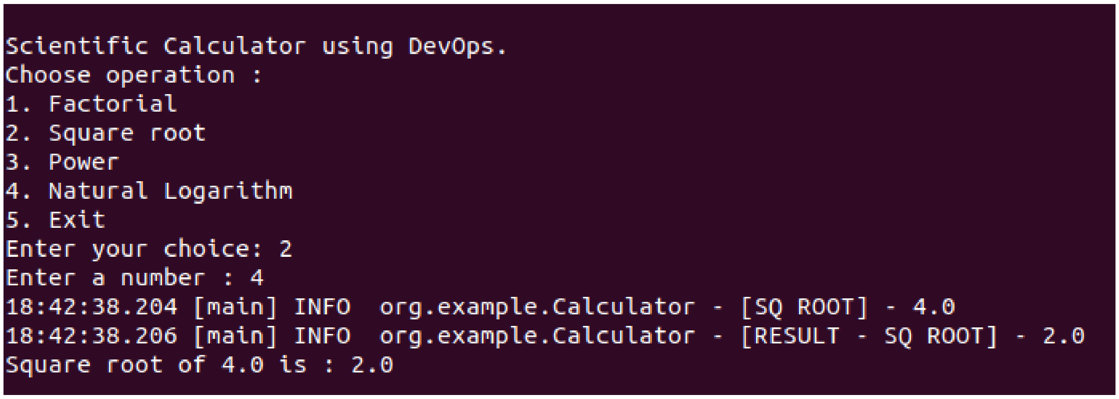


```
akanksha@akanksha-HP-Laptop-15-da1xxx:~$ docker images
REPOSITORY            TAG       IMAGE ID       CREATED          SIZE
imshukla/calculator   latest    3b1396afaa10   4 minutes ago    656MB
imshukla/calculator   <none>    a0b9fad94ae9   47 hours ago     656MB
imshukla/calculator   <none>    c791fbe65d0c   47 hours ago     656MB
imshukla/calculator   <none>    19fe4a4da17d   2 days ago       656MB
imshukla/calculator   <none>    2b4c19832be4   3 days ago       656MB
imshukla/calculator   <none>    855e55ecebd3   3 days ago       656MB
imshukla/calculator   <none>    fb36ad3cac00   3 days ago       656MB
imshukla/calculator   <none>    d1534fc8d12d   4 days ago       656MB
imshukla/calculator   <none>    c25b60b0bf32   4 days ago       654MB
imshukla/calculator   <none>    186350bad788   4 days ago       654MB
```



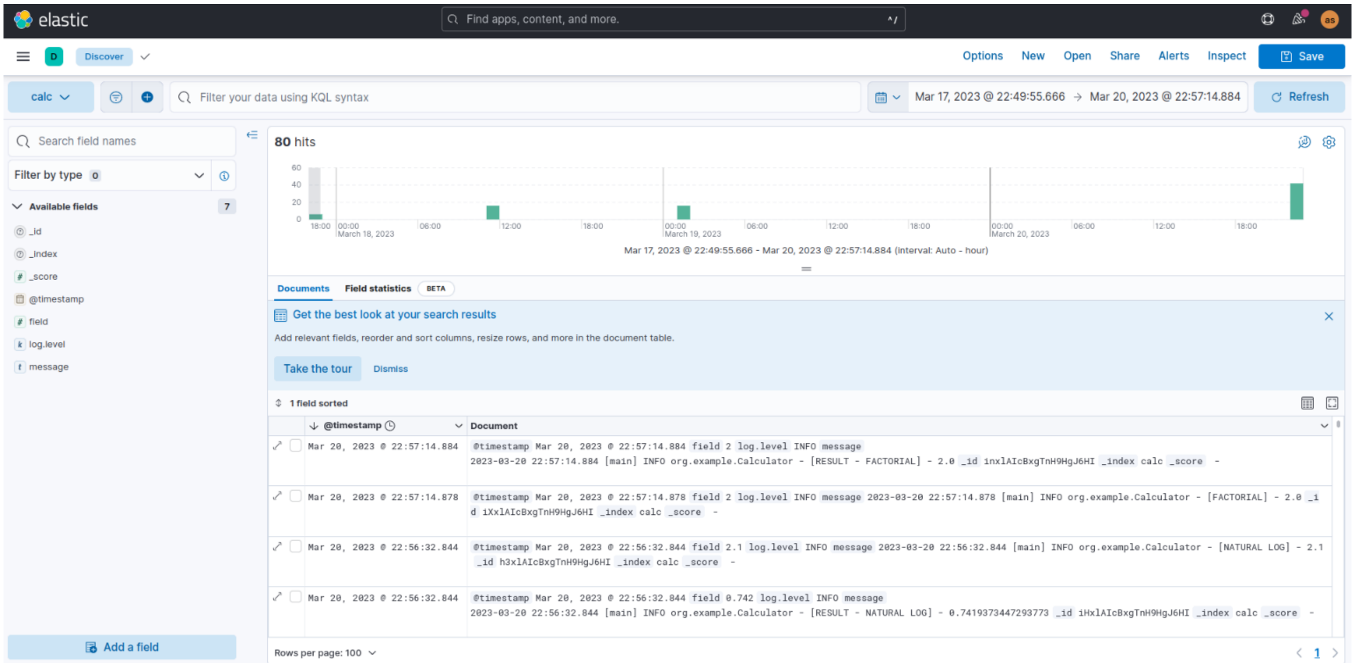
```
root@78827ab70470:/# cat calculator.log
2023-03-21 04:42:29.261 [main] INFO  org.example.Calculator - [POWER - 2.0 RAISED TO] 2.0
2023-03-21 04:42:29.265 [main] INFO  org.example.Calculator - [RESULT - POWER] - 4.0
2023-03-21 04:42:36.016 [main] INFO  org.example.Calculator - [SQ ROOT] - 4.0
2023-03-21 04:42:36.018 [main] INFO  org.example.Calculator - [RESULT - SQ ROOT] - 2.0
2023-03-21 04:42:38.847 [main] INFO  org.example.Calculator - [FACTORIAL] - 5.0
2023-03-21 04:42:38.848 [main] INFO  org.example.Calculator - [RESULT - FACTORIAL] - 120.0
2023-03-21 04:42:50.091 [main] INFO  org.example.Calculator - [FACTORIAL] - 6.0
2023-03-21 04:42:50.092 [main] INFO  org.example.Calculator - [RESULT - FACTORIAL] - 720.0
2023-03-21 04:43:03.626 [main] INFO  org.example.Calculator - [NATURAL LOG] - 4.0
2023-03-21 04:43:03.628 [main] INFO  org.example.Calculator - [RESULT - NATURAL LOG] - 1.3862943611198906
root@78827ab70470:/#
```

Calculator Output



```
Scientific Calculator using DevOps.
Choose operation :
1. Factorial
2. Square root
3. Power
4. Natural Logarithm
5. Exit
Enter your choice: 2
Enter a number : 4
18:42:38.204 [main] INFO  org.example.Calculator - [SQ ROOT] - 4.0
18:42:38.206 [main] INFO  org.example.Calculator - [RESULT - SQ ROOT] - 2.0
Square root of 4.0 is : 2.0
```

Kibana Output



@timestamp	Document
Mar 20, 2023 @ 22:57:14.884	@timestamp Mar 20, 2023 @ 22:57:14.884 field 2 log.level INFO message 2023-03-20 22:57:14.884 [main] INFO org.example.Calculator - [RESULT - FACTORIAL] - 2.0 _id inxlAIcBxgTnH9HgJ6HI _index calc _score -
Mar 20, 2023 @ 22:57:14.878	@timestamp Mar 20, 2023 @ 22:57:14.878 field 2 log.level INFO message 2023-03-20 22:57:14.878 [main] INFO org.example.Calculator - [FACTORIAL] - 2.0 _id iXxlAIcBxgTnH9HgJ6HI _index calc _score -
Mar 20, 2023 @ 22:56:32.844	@timestamp Mar 20, 2023 @ 22:56:32.844 field 2.1 log.level INFO message 2023-03-20 22:56:32.844 [main] INFO org.example.Calculator - [NATURAL LOG] - 2.1 _id h3xlAIcBxgTnH9HgJ6HI _index calc _score -
Mar 20, 2023 @ 22:56:32.844	@timestamp Mar 20, 2023 @ 22:56:32.844 field 0.742 log.level INFO message 2023-03-20 22:56:32.844 [main] INFO org.example.Calculator - [RESULT - NATURAL LOG] - 0.7419373447293773 _id iHxlAIcBxgTnH9HgJ6HI _index calc _score -